RAG LLM Evaluation and Test Automation Cheatsheet

🚀 Getting Started with RAG LLM Evaluation

>_ Set up a virtual environment

```
python3 -m venv .venv
```

>_ Activate the virtual environment

```
source .venv/bin/activate
```

>_ Install necessary libraries

```
pip install transformers sentence-transformers datasets evaluate
```

>_ Download a RAG model (e.g., from Hugging Face)

```
from transformers import pipeline; retriever = pipeline('question-answering', model='...' )
```

>_ Load your dataset (e.g., using the `datasets` library)

```
from datasets import load_dataset; dataset = load_dataset('...' )
```

⚙️ Implementing Test Cases

>_ Define a function to evaluate the RAG model's accuracy

```
def evaluate_rag(model, dataset): ...
```

>_ Create unit tests using a testing framework (e.g., pytest)

```
import pytest; def test_rag_accuracy(): ...
```

>_ Write assertions to check the correctness of the model's responses

```
assert model.predict(question) ['answer'] == expected_answer
```

>_ Implement tests for different aspects of RAG performance (e.g., latency, accuracy)

```
def test_rag_latency(): ...; def test_rag_recall(): ...
```

>_ Measure and log key metrics (e.g., F1-score, Exact Match)

```
from evaluate import load; metric = load('exact_match'); results = metric.compute(predictions=..., references=...)
```

📊 Analyzing Results and Reporting

>_ Generate a report summarizing the test results

```
import pandas as pd; pd.DataFrame(results).to_csv('report.csv')
```

>_ Visualize the results using charts or graphs

```
import matplotlib.pyplot as plt; plt.plot(...)
```

>_ Use a reporting library (e.g., pytest-html) for more comprehensive reports

```
# Install pytest-html: pip install pytest-html; Then run pytest --html=report.html
```

>_ Integrate the evaluation process into a CI/CD pipeline

```
# Example using Github Actions: Refer to Github Action documentation
```

>_ Analyze the model's performance based on various metrics (e.g., precision, recall)

```
precision = tp/(tp+fp); recall = tp/(tp+fn)
```

🤖 Advanced Techniques

>_ Implement different evaluation strategies (e.g., human evaluation)

```
# Design a survey or conduct human-in-the-loop evaluations.
```

>_ Use advanced metrics for a more nuanced evaluation

```
# Explore metrics like BLEU, ROUGE, METEOR for text generation tasks.
```

>_ Perform adversarial testing to uncover vulnerabilities in the model

```
# Generate adversarial examples to evaluate model robustness.
```

>_ Monitor model performance over time using monitoring tools

```
# Integrate with monitoring services to track model drift and degradation.
```

>_ Automate the deployment and testing of updated RAG models

```
# Automate deployment using tools like Docker and Kubernetes.
```